```
EEEEEEEEEEEEEEE  RRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR    FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR    FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

```
  SSSSSSSS  BBBBBBBB      11        11
  SSSSSSSS  BBBBBBBB      11        11
SS          BB      BB  1111      1111
SS          BB      BB  1111      1111
SS          BB      BB    11        11
SS          BB      BB    11        11
  SSSSSS    BBBBBBBB      11        11
  SSSSSS    BBBBBBBB      11        11
        SS  BB      BB    11        11
        SS  BB      BB    11        11
        SS  BB      BB    11        11      ....
        SS  BB      BB    11        11      ....
SSSSSSSS    BBBBBBBB    111111    111111    ....
SSSSSSSS    BBBBBBBB    111111    111111    ....

LL            IIIIII      SSSSSSSS
LL            IIIIII    SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0001            Subroutine SB11 (lun)
0002      C
0003      C Version:      'V04-000'
0004      C
0005      C*****************************************************************************
0006      C*                                                                           *
0007      C*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0008      C*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0009      C*  ALL RIGHTS RESERVED.                                                     *
0010      C*                                                                           *
0011      C*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0012      C*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0013      C*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0014      C*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0015      C*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0016      C*  TRANSFERRED.                                                             *
0017      C*                                                                           *
0018      C*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0019      C*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0020      C*  CORPORATION.                                                             *
0021      C*                                                                           *
0022      C*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0023      C*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0024      C*                                                                           *
0025      C*                                                                           *
0026      C*****************************************************************************
0027      C
0028      C
0029      C
0030      C       Author: Sharon Reynolds                 Creation date:  20-Jan-1981
0031      C
0032      C
0033      C       Functional description:
0034      C
0035      C       This module produces the error log report for the SB11
0036      C       DECdataway I/O subsystem.
0037      C
0038      C       Modified by:
0039      C
0040      C       V03-003 SAR0234         Sharon A. Reynolds,     28-Mar-1984
0041      C               Changed the call to UCB$L_OWNUIC to ORB$L_OWNER.
0042      C
0043      C       V03-002 SAR0110         Sharon A. Reynolds,     23-Jun-1983
0044      C               Changed the carriage control in the 'format' statements
0045      C               for use with ERF.
0046      C
0047      C       V03-001 SAR0035         Sharon A. Reynolds,     8-Jun-1983
0048      C               Removed brief/cryptic support.
0049      C
0050      C       v02-004 BP0004          Brian Porter,           23-NOV-1981
0051      C               Minor edit.
0052      C
0053      C       v02-003 BP0003          Brian Porter,           05-NOV-1981
0054      C               Added 'device attention' support.
0055      C
0056      C       v02-002 BP0002          Brian Porter,           23-JUL-1981
0057      C               Added new uba routines.
```

```
0058    C
0059    C       v02-001 BP0001          Brian Porter,           15-JUL-1981
0060    C               Misc. changes made by S. Reynolds in response to input
0061    C               from W. Saltz.  Added call to DHEAD1.
0062    C**
0063
0064            Include 'SRC$:MSGHDR.FOR /NOLIST'
0123            Include 'SRC$:DEVERR.FOR /NOLIST'
0224
0225            Byte            lun
0226            Byte            hdr_map_valid
0227            Byte            dat_map_valid
0228            Byte            att_ucb0
0229            Byte            port_id
0230
0231            Integer*2       csr0
0232            Integer*2       csr1
0233            Integer*2       csr2
0234
0235            Integer*4       COMPRESS4
0236            Integer*4       COMPRESSC
0237            Integer*4       no_irp
0238            Integer*4       field
0239            Integer*4       unit
0240            Integer*4       devdepend
0241            Integer*4       devsts
0242            Integer*4       tt_retry_cnt
0243            Integer*4       tt_protocol_err
0244            Integer*4       tumble_table
0245            Integer*4       registers
0246            Integer*4       err_sts
0247            Integer*4       uba_reg1(0:2)
0248            Integer*4       uba_reg2(0:2)
0249            Integer*4       dataway_addr
0250
0251            Parameter       timeout = 96
0252            Parameter       rt80 = 1
0253            Parameter       dyt01 = 2
0254            Parameter       dpm01 = 4
0255            Parameter       dis = 8
0256
0257            Character*83    tt_protocol_id(0:7)
0258            Character*15    tumble_tbl(5:7)
0259            Character*35    err_num(0:31)
0260            Character*29    func(0:7)
0261            Character*22    csr0_1(5:7)
0262            Character*32    csr2_1(6:7)
0263            Character*6     device(1:8)
0264            Character*46    dvsts(0:11)
0265
0266    C
0267    C       Make the register save area of the error log buffer available
0268    C       to this module
0269    C
0270
0271            Equivalence     (err_sts,EMB$L_DV_REGSAV(0))
0272            Equivalence     (registers,EMB$L_DV_REGSAV(1))
```

```
0273            Equivalence     (devdepend,EMB$L_DV_REGSAV(2))
0274            Equivalence     (devsts,EMB$L_DV_REGSAV(3))
0275            Equivalence     (uba_reg1,EMB$L_DV_REGSAV(4))
0276            Equivalence     (uba_reg2,EMB$L_DV_REGSAV(7))
0277
0278    C
0279    C       Define text for the tumble table
0280    C
0281
0282            Data    tt_protocol_id(0)       /'PREMATURE END OF MESSAGE'/
0283            Data    tt_protocol_id(1)       /'BAD ADDRESS'/
0284            Data    tt_protocol_id(2)       /'ILLEGAL RESPONSE CONTROL CODE'/
0285            Data    tt_protocol_id(3)       /'SEQUENCE BIT WRONG-NO REPEAT!COMMAND SENT'/
0286            Data    tt_protocol_id(4)       /'ILLEGAL MSG-LENGTH ON!NON ''SI'' RESPONSE'/
0287            Data    tt_protocol_id(5)       /'ILLEGAL RESPONSE OF ''SI''!WHEN NOT
0288            1 REQUESTED'/
0289            Data    tt_protocol_id(6)       /'ILLEGAL RESPONSE OF ''RNR'' TO!''RI''
0290            1 COMMAND WHEN READY TO!RECEIVE NOT REQUESTED'/
0291            Data    tt_protocol_id(7)       /'ILLEGAL RESPONSE OF ''RNR'' TO!''RPS''
0292            1 COMMAND WHEN READY TO!RECEIVE NOT REQUIRED'/
0293
0294            Data    tumble_tbl(5)   /'PROTOCOL ERROR*'/
0295            Data    tumble_tbl(6)   /'TIMEOUT ERROR*'/
0296            Data    tumble_tbl(7)   /'CRC ERROR*'/
0297
0298
0299    C
0300    C       Define text for error number
0301    C
0302
0303            Data    err_num(0)      /'LATENCY ERROR-NO DATA!TRANSFERRED'/
0304            Data    err_num(1)      /'LATENCY ERROR-DATA TRANSFERRED'/
0305            Data    err_num(2)      /'MEMORY TIMEOUT ERROR'/
0306            Data    err_num(3)      /'ONLINE TRANSITION'/
0307            Data    err_num(4)      /'OFFLINE TRANSITION'/
0308            Data    err_num(5)      /'DATA OVERRUN ERROR'/
0309            Data    err_num(6)      /'WRITE FUNCTION TIMEOUT'/
0310            Data    err_num(7)      /'UNSOLICITED DATA'/
0311            Data    err_num(8)      /'LINE ERROR'/
0312            Data    err_num(9)      /'RESERVED'/
0313            Data    err_num(10)     /'NO DATA TRANSFERRED'/
0314            Data    err_num(11)     /'INVALID TYPE 2 TUMBLE!TABLE ENTRY'/
0315            Data    err_num(12)     /'I/O DONE WITH NO TRANSFER'/
0316            Data    err_num(13)     /'ONLINE I/O DONE'/
0317            Data    err_num(14)     /'INVALID DEVICE UNIT NUMBER'/
0318            Data    err_num(15)     /'NON-EXISTENT MEMORY'/
0319            Data    err_num(16)     /'PORT WAIT QUEUE TIMEOUT'/
0320            Data    err_num(17)     /'USER INITIATED SHUTDOWN'/
0321            Data    err_num(18)     /'CONTROLLER STARTUP'/
0322            Data    err_num(19)     /'INVALID TYPE 1 TUMBLE!TABLE ENTRY'/
0323            Data    err_num(20)     /'DRIVER DATA STRUCTURE ERROR'/
0324            Data    err_num(21)     /'RESERVED'/
0325            Data    err_num(22)     /'RESERVED'/
0326            Data    err_num(23)     /'RESERVED'/
0327            Data    err_num(24)     /'AST NOT DELIVERABLE'/
0328            Data    err_num(25)     /'ENABLE LOGGING OF DECDATAWAY!ERROR'/
0329            Data    err_num(26)     /'DISABLE LOGGING OF DECDATAWAY!ERROR'/
```

```
0330            Data    err_num(27)     /'RESERVED'/
0331            Data    err_num(28)     /'RESERVED'/
0332            Data    err_num(29)     /'DRIVER INITIATED SHUTDOWN'/
0333            Data    err_num(30)     /'RESERVED'/
0334            Data    err_num(31)     /'RESERVED'/
0335
0336
0337    C
0338    C       Define text for bits in CSR0
0339    C
0340
0341            Data    func(0)         /'NO-OP*'/
0342            Data    func(1)         /'INITIATE TRANSFER OUT*'/
0343            Data    func(2)         /'INITIATE TRANSFER IN*'/
0344            Data    func(3)         /'SEND TRANSPARENT DATA*'/
0345            Data    func(4)         /'SEND TRANSPARENT DATA-NO CRC*'/
0346            Data    func(5)         /'RESERVED*'/
0347            Data    func(6)         /'RESERVED*'/
0348            Data    func(7)         /'CANCEL OPERATION*'/
0349
0350            Data    csr0_1(5)       /'HARDWARE ERROR*'/
0351            Data    csr0_1(6)       /'DONE INTERRUPT ENABLE*'/
0352            Data    csr0_1(7)       /'DONE*'/
0353
0354
0355    C
0356    C       Define text for bits in CSR2
0357    C
0358
0359            Data    csr2_1(6)       /'PORT AVAILABLE INTERRUPT ENABLE*'/
0360            Data    csr2_1(7)       /'PORT LOCK*'/
0361
0362
0363    C
0364    C       Define the devices currently allowed on the DECdataway
0365    C
0366
0367            Data    device(1)       /'RT80*'/
0368            Data    device(2)       /'DYT01*'/
0369            Data    device(4)       /'DPM01*'/
0370            Data    device(8)       /'DIS*'/
0371
0372    C
0373    C       Define the text for the bit assignments in ucb$w_devsts
0374    C
0375
0376            Data    dvsts(0)        /'REQUEST IN THE PORT-WAIT-QUEUE*'/
0377            Data    dvsts(1)        /'CANCEL IN PROGRESS*'/
0378            Data    dvsts(2)        /'TRANSFER OPERATION IS REQUESTED*'/
0379            Data    dvsts(3)        /'UNIT IS ACTIVE*'/
0380            Data    dvsts(4)        /'SYSTEM ERROR LOGGING ENABLED*'/
0381            Data    dvsts(5)        /'LOGGING OF ON/OFF LINE ENABLED*'/
0382            Data    dvsts(6)        /'USER ERROR LOG IS ENABLED*'/
0383            Data    dvsts(7)        /'INHIBIT ERROR LOGGING*'/
0384            Data    dvsts(8)        /'UNIT IS SHUTDOWN*'/
0385            Data    dvsts(9)        /'REQUEST HAS TIMED OUT*'/
0386            Data    dvsts(10)       /'RESERVED*'/
```

```
0387            Data    dvsts(11)       /'REQUEST ONLINE*'/
0388
0389
0390    C
0391    C       Construct the error log report header
0392    C
0393
0394            Call FRCTOF (lun)
0395
0396            call dhead1 (lun,'UBA DEC DATAWAY')
0397
0398    C
0399    C       Extract necessary information for later use
0400    C
0401
0402            Hdr_map_valid = LIB$EXTZV (30,1,EMB$L_DV_NUMREG)
0403
0404            Dat_map_valid = LIB$EXTZV (31,1,EMB$L_DV_NUMREG)
0405
0406            No_irp = LIB$EXTZV (0,16,EMB$L_DV_NUMREG)
0407
0408    C
0409    C       Decode and output the bits in the csr0 register
0410    C
0411
0412            Csr0 = LIB$EXTZV (0,8,registers)
0413
0414            Call LINCHK (lun,2)
0415            Write (lun,30) csr0
0416    30      Format (/' ',T8,'CSR0',T24,Z8.8)
0417
0418            Field = LIB$EXTZV (0,3,csr0)
0419
0420            Call LINCHK (lun,1)
0421            Write (lun,40) func(field)
0422    40      Format (' ',T40,'FUNCTION = ',A<COMPRESSC (func(field))> )
0423
0424            Call OUTPUT (lun,csr0,csr0_1,5,5,7,'0')
0425
0426    C
0427    C       Decode and output the bits in the csr1 register
0428    C
0429
0430            Csr1 = LIB$EXTZV (8,8,registers)
0431
0432            Call LINCHK (lun,2)
0433            Write (lun,50) csr1
0434    50      Format (/' ',T8,'CSR1',T24,Z8.8)
0435
0436
0437
0438    C
0439    C       Decode and output the bits in the csr2 register
0440    C
0441
0442            Csr2 = LIB$EXTZV (16,8,registers)
0443
```

```
0444            Call LINCHK (lun,2)
0445            Write (lun,55) csr2
0446    55      Format (/' ',T8,'CSR2',T24,Z8.8)
0447
0448            Dataway_addr = LIB$EXTZV (0,6,csr2)
0449
0450            Call LINCHK (lun,1)
0451            Write (lun,60) dataway_addr
0452    60      Format (' ',T40,'DECDATAWAY ADDRESS = ',
0453           1 I<COMPRESS4 (dataway_addr)>,'.')
0454
0455            Call OUTPUT (lun,csr2,csr2_1,6,6,7,'0')
0456
0457    C
0458    C       Decode and output the mapping information for the header and/or
0459    C       data buffers
0460    C
0461
0462            If (no_irp .eq. 10) then
0463
0464            If (hdr_map_valid .eq. 1) then
0465
0466            Call LINCHK (lun,3)
0467            Write (lun,70)
0468    70      Format (/' ','HEADER BUFFER MAPPING INFORMATION',/)
0469
0470            call uba_mapping (lun,-1,uba_reg1(0))
0471
0472            call uba_mapping (lun,-1,uba_reg1(1))
0473
0474            call vecmapreg (lun,uba_reg1(2))
0475            endif
0476
0477            If (dat_map_valid .eq. 1) then
0478
0479            Call LINCHK (lun,2)
0480            Write (lun,80)
0481    80      Format (/' ','DATA BUFFER MAPPING INFORMATION')
0482
0483            call uba_mapping (lun,-1,uba_reg2(0))
0484
0485            call uba_mapping (lun,-1,uba_reg2(1))
0486
0487            call vecmapreg (lun,uba_reg2(2))
0488            Endif
0489            Endif
0490
0491    C
0492    C       Decode and output the 'dataway' protocol and error status
0493    C       information
0494    C
0495
0496            Call LINCHK (lun,2)
0497            Write (lun,90)
0498    90      Format (/' ','DECDATAWAY PROTOCOL AND STATUS INFORMATION')
0499
0500            Call LINCHK (lun,2)
```

```
0501            Write (lun,100) err_sts
0502    100     Format (/' ',T8,'ERROR STATUS',T24,Z8.8)
0503
0504            Field = LIB$EXTZV (0,5,err_sts)
0505
0506            Call OUTPUT_MLINES (lun,err_num(field),'!',32)
0507
0508    C
0509    C       Find out if the tumble table entry is valid and output it when
0510    C       necessary
0511    C
0512
0513            If (field .eq. 7
0514            1 .or.
0515            2 field  .eq. 8) then
0516
0517            Call LINCHK (lun,1)
0518            Write (lun,110)
0519    110     Format (' ',T40,'********** TUMBLE TABLE **********')
0520
0521            Tumble_table = LIB$EXTZV (16,8,err_sts)
0522
0523            If (field .eq. 8) then
0524
0525            Tt_retry_cnt = LIB$EXTZV (0,2,tumble_table)
0526
0527            Call LINCHK (lun,2)
0528            Write (lun,120) tt_retry_cnt + 1
0529    120     Format (' ',T40,'REPEAT COUNT = ',
0530            1 I<COMPRESS4 (tt_retry_cnt)>,'.')
0531            Write (lun,121)
0532    121     Format (' ')
0533
0534            Call OUTPUT (lun,tumble_table,tumble_tbl,5,5,7,'0')
0535
0536            Tt_protocol_err = LIB$EXTZV (5,1,tumble_table)
0537
0538            If (tt_protocol_err .eq. 1) then
0539
0540            Field = LIB$EXTZV (2,3,tumble_table)
0541
0542            Call OUTPUT_MLINES (lun,tt_protocol_id(field),'!',32)
0543            Endif
0544            Endif
0545
0546            If (field .eq. 7) then
0547
0548            Call LINCHK (lun,2)
0549            Write (lun,130) tumble_table
0550    130     Format (/' ',T40, 'DATA = ',Z4.4)
0551
0552            Endif
0553
0554            Call LINCHK (lun,1)
0555            Write (lun,140) ('*', I=1,34)
0556    140     Format (' ',T40,34A1)
0557            Endif
```

```
0558
0559            Unit = LIB$EXTZV (5,6,err_sts)
0560
0561            Call LINCHK (lun,1)
0562            Write (lun,150) unit
0563    150     Format (' ',T40,'UNIT = ',I<COMPRESS4 (unit)>,'.')
0564
0565    C
0566    C       Decode the 'SOFTWARE INFORMATION' and return to ERRPRT
0567    C
0568    C       If there was no I/O pending, not all of the data included under
0569    C       software information will be valid, so only output the software
0570    C       information that is valid.
0571    C
0572
0573            call linchk (lun,1)
0574
0575            write(lun,155)
0576    155     format(' ',:)
0577
0578            if (no_irp .ne. 4) then
0579
0580            if (emb$w_hd_entry .ne. 98) then
0581
0582            call ucb$b_ertcnt (lun,emb$b_dv_ertcnt)
0583
0584            call ucb$b_ertmax (lun,emb$b_dv_ertmax)
0585            endif
0586            endif
0587
0588            call orb$l_owner (lun,emb$l_dv_ownuic)
0589
0590            call ucb$l_char (lun,emb$l_dv_char)
0591
0592            call ucb$w_sts (lun,emb$w_dv_sts)
0593
0594            call ucb$l_opcnt (lun,emb$l_dv_opcnt)
0595
0596            call ucb$w_errcnt (lun,emb$w_dv_errcnt)
0597
0598            Att_ucb0 = LIB$EXTZV (31,1,devdepend)
0599            Field = LIB$EXTZV (4,4,devdepend)
0600            Port_id = LIB$EXTZV (0,4,devdepend)
0601
0602            Call LINCHK (lun,2)
0603            Write (lun,160) devdepend
0604    160     Format (/' ',T8,'UCB$L_DEVDEPEND',T24,Z8.8)
0605
0606    C
0607    C       The port id for the RT80x indicates whether the device is
0608    C       an RT801, RT803, RT805. Determine if the device is an RT80x
0609    C       and if so set up the port id to the proper value to output
0610    C
0611            Call LINCHK (lun,1)
0612            If (field .eq. rt80) then
0613
0614            If (port_id .eq. 2) then
```

```
0615            Port_id = port_id + 1
0616
0617            Else if (port_id .eq. 3) then
0618            Port_id = port_id + 2
0619            Endif
0620
0621            Write (lun,165) device(field),port_id
0622    165     Format (' ',T40,A<COMPRESSC (device(field))>,I1)
0623
0624
0625            Else if (field .eq. dyt01) then
0626    C
0627    C       Device is a DYT01 , it has only one port so output the
0628    C       device type
0629    C
0630            Write (lun,167) device(field)
0631    167     Format (' ',T40,A<COMPRESSC (device(field))>)
0632
0633
0634            Else if (field .eq. dpm01
0635            1  .or.
0636            2  field .eq. dis) then
0637    C
0638    C       The device is either a DPM01 or Distributed Intelligent Subsystem,
0639    C       both currently support four ports (0-3). Output the device type
0640    C       and the port identification
0641    C
0642            Write (lun,170) device(field),port_id
0643    170     Format (' ',T40,A<COMPRESSC (device(field))>,
0644            1 ' - PORT #',I1)
0645            endif
0646
0647            If (att_ucb0 .eq. 1) then
0648
0649            Call LINCHK (lun,1)
0650            Write (lun,175)
0651    175     Format (' ',T40,'ATTENTION UCB0')
0652
0653            Endif
0654
0655            Call LINCHK (lun,2)
0656            Write (lun,180) devsts
0657    180     Format (/' ',T8,'UCB$W_DEVSTS',T24,Z8.8)
0658
0659            Call OUTPUT (lun,devsts,dvsts,0,0,11,'0')
0660
0661            if (no_irp .ne. 4) then
0662
0663            if (emb$w_hd_entry .ne. 98) then
0664
0665            call linchk (lun,1)
0666
0667            write(lun,155)
0668
0669            call sb11_qio (lun,emb$w_dv_func)
0670
0671            call irp$w_bcnt (lun,emb$w_dv_bcnt)
```

```
0672
0673            call irp$w_boff (lun,emb$w_dv_boff)
0674
0675            call irp$l_pid (lun,emb$l_dv_rqpid)
0676
0677            call irp$q_iosb (lun,emb$l_dv_iosb1)
0678            endif
0679            endif
0680
0681            Return
0682            End
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	2095	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	616	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	3804	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	EMB	512	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	7027	

ENTRY POINTS

Address	Type	Name
0-00000000		SB11

VARIABLES

Address	Type	Name	Address	Type	Name
2-00000AE9	L*1	ATT_UCB0	2-00000AEC	I*2	CSR0
2-00000AEE	I*2	CSRT	2-00000AF0	I*2	CSR2
2-00000B0C	I*4	DATAWAY_ADDR	2-00000AE8	L*1	DAT_MAP_VALID
3-0000005A	I*4	DEVDEPEND	3-0000005E	I*4	DEVSTS
3-0000001C	L*1	EMB$B_DV_CLASS	3-00000010	L*1	EMB$B_DV_ERTCNT
3-00000011	L*1	EMB$B_DV_ERTMAX	3-0000003E	L*1	EMB$B_DV_NAMLNG
3-0000003A	L*1	EMB$B_DV_SLAVE	3-0000001D	L*1	EMB$B_DV_TYPE
3-00000036	I*4	EMB$L_DV_CHAR	3-00000012	I*4	EMB$L_DV_IOSB1
3-00000016	I*4	EMB$L_DV_IOSB2	3-00000026	I*4	EMB$L_DV_MEDIA
3-0000004E	I*4	EMB$L_DV_NUMREG	3-0000002E	I*4	EMB$L_DV_OPCNT
3-00000032	I*4	EMB$L_DV_OWNUIC	3-0000001E	I*4	EMB$L_DV_RQPID
3-00000000	I*4	EMB$L_HD_SID	3-0000003F	CHAR	EMB$T_DV_NAME
3-00000024	I*2	EMB$W_DV_BCNT	3-00000022	I*2	EMB$W_DV_BOFF
3-0000002C	I*2	EMB$W_DV_ERRCNT	3-0000003C	I*2	EMB$W_DV_FUNC
3-0000001A	I*2	EMB$W_DV_STS	3-0000002A	I*2	EMB$W_DV_UNIT
3-00000004	I*2	EMB$W_HD_ENTRY	3-0000000E	I*2	EMB$W_HD_ERRSEQ
3-00000052	I*4	ERR_STS	2-00000AF8	I*4	FIELD
2-00000AE7	L*1	HDR_MAP_VALID	2-00000B10	I*4	I
AP-00000004@	L*1	LUN	2-00000AF4	I*4	NO_IRP

2-00000AEA	L*1	PORT_ID	3-00000056	I*4	REGISTERS
2-00000B04	I*4	TT_PROTOCOL_ERR	2-00000B00	I*4	TT_RETRY_CNT
2-00000B08	I*4	TUMBLE_TABLE	2-00000AFC	I*4	UNIT

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-0000080D	CHAR	CSR0_1	66	(5:7)
2-0000084F	CHAR	CSR2_1	64	(6:7)
2-0000088F	CHAR	DEVICE	48	(8)
2-000008BF	CHAR	DVSTS	552	(0:11)
3-00000000	L*1	EMB	512	(0:511)
3-00000052	I*4	EMB$L_DV_REGSAV	420	(0:104)
3-00000006	I*4	EMB$Q_HD_TIME	8	(2)
2-000002C5	CHAR	ERR_NUM	1120	(0:31)
2-00000725	CHAR	FUNC	232	(0:7)
2-00000000	CHAR	TT_PROTOCOL_ID	664	(0:7)
2-00000298	CHAR	TUMBLE_TBL	45	(5:7)
3-00000062	I*4	UBA_REG1	12	(0:2)
3-0000006E	I*4	UBA_REG2	12	(0:2)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
1-00000051	30'	1-00000063	40'	1-0000007C	50'	1-0000008E	55'	1-000000A0	60'	1-000000C6	70'
1-000000EF	80'	1-00000115	90'	1-00000146	100'	1-00000160	110'	1-0000018A	120'	1-000001AA	121'
1-000001AE	130'	1-000001C1	140'	1-000001CB	150'	1-000001E3	155'	1-000001E8	160'	1-00000205	165'
1-00000213	167'	1-0000021F	170'	1-00000238	175'	1-0000024E	180'				

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name	Type	Name	Type	Name	Type	Name
I*4	COMPRESS4	I*4	COMPRESSC		DHEAD1		FRCTOF		IRP$L_PID		IRP$Q_IOSB
	IRP$W_BCNT		IRP$W_BOFF	I*4	LIB$EXTZV		LINCHK		ORB$L_OWNER		OUTPUT
	OUTPUT_MLINES		SB11_QIO		UBA_MAPPING		UCB$B_ERTCNT		UCB$B_ERTMAX		UCB$L_CHAR
	UCB$L_OPCNT		UCB$W_ERRCNT		UCB$W_STS		VECMAPREG				

```
0001
0002
0003
0004          Subroutine SB11_QIO (lun,emb$w_dv_func)
0005
0006
0007          include 'src$:qiocommon.for /nolist'
0271
0272
0273          byte            lun
0274
0275          integer*2       emb$w_dv_func
0276
0277          integer*4       qiocode(0:1,0:63)
0278
0279
0280
0281
0282          if (qiocode(0,0) .eq. 0) then
0283
0284          Qiocode(1,24)   = %loc(IO$_WRITECSR)
0285
0286          Qiocode(1,25)   = %loc(IO$_READCSR)
0287
0288          Qiocode(1,26)   = %loc(IO$_SETCHAR)
0289
0290          Qiocode(1,32)   = %loc(IO$_WRITELBLK)
0291
0292          Qiocode(1,33)   = %loc(IO$_READLBLK)
0293
0294          Qiocode(1,34)   = %loc(IO$_ABORT)
0295
0296          Qiocode(1,35)   = %loc(IO$_SETMODE)
0297
0298          Qiocode(1,36)   = %loc(IO$_WRITEWTHBUF)
0299
0300          Qiocode(1,37)   = %loc(IO$_READWTHBUF)
0301
0302          Qiocode(1,39)   = %loc(IO$_SENSEMODE)
0303
0304          Qiocode(1,40)   = %loc(IO$_WRITEBUFNCRC)
0305
0306          Qiocode(1,41)   = %loc(IO$_READWTHXBUF)
0307
0308          Qiocode(1,48)   = %loc(IO$_WRITEVBLK)
0309
0310          Qiocode(1,49)   = %loc(IO$_READVBLK)
0311
0312          do 10,i = 0,63
0313
0314          qiocode(0,i) = 33
0315
0316          if (qiocode(1,i) .eq. 0) then
0317
0318          qiocode(1,i) = %loc(qio_string)
0319          endif
0320
```

```
0321     10      continue
0322             endif
0323
0324             call irp$w_func (lun,emb$w_dv_func,
0325             1 qiocode(0,lib$extzv(0,6,emb$w_dv_func)))
0326
0327             return
0328
0329             end
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	197	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	8	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	548	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	QIOCOMMON	1247	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	2000	

ENTRY POINTS

Address	Type	Name
0-00000000		SB11_QIO

VARIABLES

Address	Type	Name	Address	Type	Name
AP-0000008a	I*2	EMB$W_DV_FUNC	2-00000200	I*4	I
3-00000442	CHAR	IO$_ABORT	3-0000034D	CHAR	IO$_ACCESS
3-000003C2	CHAR	IO$_ACPCONTROL	3-000004B3	CHAR	IO$_AVAILABLE
3-00000297	CHAR	IO$_CLEAN	3-00000369	CHAR	IO$_CREATE
3-00000385	CHAR	IO$_DEACCESS	3-00000393	CHAR	IO$_DELETE
3-0000026D	CHAR	IO$_DIAGNOSE	3-00000065	CHAR	IO$_DRVCLR
3-000004CB	CHAR	IO$_DSE	3-000000A9	CHAR	IO$_ERASETAPE
3-00000276	CHAR	IO$_FORMAT	3-00000071	CHAR	IO$_INITIALIZE
3-00000014	CHAR	IO$_LOADMCODE	3-000003A1	CHAR	IO$_MODIFY
3-000003E2	CHAR	IO$_MOUNT	3-00000000	CHAR	IO$_NOP
3-0000009D	CHAR	IO$_OFFSET	3-000000EB	CHAR	IO$_PACKACK
3-000000E0	CHAR	IO$_QSTOP	3-000003EF	CHAR	IO$_RDSTATS
3-00000421	CHAR	IO$_READCSR	3-00000169	CHAR	IO$_READHEAD
3-000002B6	CHAR	IO$_READLBLK	3-0000013F	CHAR	IO$_READPBLK
3-00000200	CHAR	IO$_READPRESET	3-00000195	CHAR	IO$_READTRACKD
3-0000033A	CHAR	IO$_READVBLK	3-0000045A	CHAR	IO$_READWTHBUF
3-00000484	CHAR	IO$_READWTHXBUF	3-0000004D	CHAR	IO$_RECAL
3-0000007C	CHAR	IO$_RELEASE	3-000001AB	CHAR	IO$_REREADN
3-000001B8	CHAR	IO$_REREADP	3-000000CA	CHAR	IO$_RETCENTER
3-000002E6	CHAR	IO$_REWIND	3-000002C9	CHAR	IO$_REWINDOFF
3-000000FC	CHAR	IO$_SEARCH	3-00000024	CHAR	IO$_SEEK

```
  3-00000231  CHAR IO$_SENSECHAR          3-00000309  CHAR IO$_SENSEMODE
  3-0000021D  CHAR IO$_SETCHAR            3-000003B8  CHAR IO$_SETCLOCK
  3-00000088  CHAR IO$_SETCLOCKP          3-000002DD  CHAR IO$_SETMODE
  3-000002ED  CHAR IO$_SKIPFILE           3-000002FA  CHAR IO$_SKIPRECORD
  3-00000029  CHAR IO$_SPACEFILE          3-0000010E  CHAR IO$_SPACERECORD
  3-000003D7  CHAR IO$_STARTDATA          3-000000B4  CHAR IO$_STARTDATAP
  3-00000037  CHAR IO$_STARTMPROC         3-0000020F  CHAR IO$_STARTSPNDL
  3-00000059  CHAR IO$_STOP               3-0000000D  CHAR IO$_UNLOAD
  3-0000046B  CHAR IO$_WRITEBUFNCRC       3-0000011E  CHAR IO$_WRITECHECK
  3-000001E4  CHAR IO$_WRITECHECKH        3-000003FF  CHAR IO$_WRITECSR
  3-00000153  CHAR IO$_WRITEHEAD          3-000002A2  CHAR IO$_WRITELBLK
  3-00000247  CHAR IO$_WRITEMARK          3-00000314  CHAR IO$_WRITEOF
  3-0000012A  CHAR IO$_WRITEPBLK          3-000001C9  CHAR IO$_WRITERET
  3-0000017E  CHAR IO$_WRITETRACKD        3-00000326  CHAR IO$_WRITEVBLK
  3-00000448  CHAR IO$_WRITEWTHBUF        3-00000257  CHAR IO$_WRTTMKR
 AP-0000004A  L*1  LUN                    3-000004A1  CHAR QIO_STRING
```

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-00000000	I*4	QIOCODE	512	(0:1, 0:63)

LABELS

Address	Label
**	10

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name
	IRP$W_FUNC	I*4	LIB$EXTZV

COMMAND QUALIFIERS

```
  FORTRAN /LIS=LIS$:SB11/OBJ=OBJ$:SB11 MSRC$:SB11

  /CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /STANDARD=(NOSYNTAX,NOSOURCE_FORM)
  /SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOCROSS_REFERENCE  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

```
  Run Time:                8.68 seconds
  Elapsed Time:           18.05 seconds
  Page Faults:              232
  Dynamic Memory:           221 pages
```

RECSELECT
LIS

RLDISK
LIS

PUDRIVER
LIS

PCL11T
LIS

ROLLUP
LIS

RXDISK
LIS

PCL11R
LIS

SB11
LIS

RKDISK
LIS

TIMCMP
LIS

TUTAPE
LIS

SBI
LIS

STSEVENT
LIS

TRNS_BITS
LIS

UBAERR
LIS

TIMRB
LIS

UBA
LIS

TU81SENSE
LIS

UNDEFINED
LIS

SYSPWRFL
LIS

SUMMARY
LIS

UNKN_DISP
LIS

SHRVECTOR
LIS

SYSTARTUP
LIS

UBAINT
LIS

TOF
LIS

UNKNOWN
LIS

TSTAPE
LIS